import numpy as np import pandas as pd import keras import cv2 import datetime import random as rn import tensorflow as tf from tensorflow.keras.models import * from tensorflow.keras.layers import * from tensorflow.keras.preprocessing.text import Tokenizer from tensorflow.keras.preprocessing.sequence import pad_sequences from tensorflow.keras.models import Sequential, Model from tensorflow.keras.layers import Dense, MaxPooling1D, MaxPooling2D from tensorflow.keras.layers import Flatten, Dropout, concatenate, Input from tensorflow.keras.layers import Embedding from tensorflow.keras.layers import Conv1D, Conv2D from tensorflow.keras.callbacks import TensorBoard,Callback from tensorflow.keras.callbacks import EarlyStopping %reload_ext tensorboard from tensorflow.keras.utils import plot_model from keras_preprocessing.image import ImageDataGenerator import pydot_ng as pydot pydot.find_graphviz() In [4]: images = pd.read_csv('labels_final.csv') images['label'] = images['label'].astype(str) images_directory = 'data_final/' In [5]: $target_size = (128, 128)$ In [6]: datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2) In [7]: train_generator = datagen.flow_from_dataframe(images, images_directory, x_col='path',y_col='label',class_mode='categorical', target_size=target_size, batch_si Found 38400 validated image filenames belonging to 16 classes. In [8]: validation_generator = datagen.flow_from_dataframe(images, images_directory, x_col='path',y_col='label',class_mode='categorical', target_size=target_size, ba Found 9600 validated image filenames belonging to 16 classes. In [33]: tf.keras.backend.clear_session() np.random.seed(0) rn.seed(0) base_model = tf.keras.applications.vgg16.VGG16(weights='imagenet', include_top=False, input_shape=(*target_size, 3)) for layer in base_model.layers: layer.trainable = False Conv2D_1 = Conv2D(filters = 512, kernel_size=(3, 3), activation='relu')(base_model.output) Pool_1 = MaxPooling2D()(Conv2D_1) flatten = Flatten(data_format='channels_last', name='Flatten')(Pool_1) dense_1 = Dense(units = 2048, activation='relu')(flatten) dense_2 = Dense(units = 512, activation='relu')(dense_1) = Dense(units = 16, activation='softmax')(dense_2) model_1 = Model(inputs=base_model.inputs, outputs=Out) model_1.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy']) In [14]: log_dir ="logs/fit/"+"model_1_" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S") tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,histogram_freq=1, write_graph=True,write_grads=True) early_stoping = keras.callbacks.EarlyStopping(patience=2) WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback. In [15]: class AchieveAccuracy(Callback): def __init__(self, target): super(AchieveAccuracy, self).__init__() self.target = target def on_epoch_end(self, epoch, logs={}): acc = logs['accuracy'] if acc >= self.target: self.model.stop_training = True In [16]: %%time $batch_size = 256$ steps = len(train_generator.labels) // batch_size target = AchieveAccuracy(0.6) callbacks = [early_stoping, tensorboard_callback, target] print(model_1.summary()) Model: "model" Output Shape Param # Layer (type) input_1 (InputLayer) [(None, 128, 128, 3)] block1_conv1 (Conv2D) (None, 128, 128, 64) 1792 block1_conv2 (Conv2D) (None, 128, 128, 64) 36928 block1_pool (MaxPooling2D) (None, 64, 64, 64) block2_conv1 (Conv2D) (None, 64, 64, 128) 73856 block2_conv2 (Conv2D) (None, 64, 64, 128) 147584 block2_pool (MaxPooling2D) (None, 32, 32, 128) block3_conv1 (Conv2D) (None, 32, 32, 256) 295168 block3_conv2 (Conv2D) (None, 32, 32, 256) 590080 block3_conv3 (Conv2D) (None, 32, 32, 256) 590080 block3_pool (MaxPooling2D) (None, 16, 16, 256) block4_conv1 (Conv2D) (None, 16, 16, 512) 1180160 block4_conv2 (Conv2D) (None, 16, 16, 512) 2359808 block4_conv3 (Conv2D) (None, 16, 16, 512) 2359808 block4_pool (MaxPooling2D) (None, 8, 8, 512) block5_conv1 (Conv2D) 2359808 (None, 8, 8, 512) block5_conv2 (Conv2D) 2359808 (None, 8, 8, 512) block5_conv3 (Conv2D) (None, 8, 8, 512) 2359808 block5_pool (MaxPooling2D) (None, 4, 4, 512) conv2d (Conv2D) (None, 2, 2, 512) 2359808 max_pooling2d (MaxPooling2D) (None, 1, 1, 512) 0 Flatten (Flatten) (None, 512) 0 dense (Dense) (None, 2048) 1050624 dense_1 (Dense) (None, 512) 1049088 dense_2 (Dense) (None, 16) 8208 Total params: 19,182,416 Trainable params: 4,467,728 Non-trainable params: 14,714,688 None Wall time: 6.95 ms In [26]: history_1 = model_1.fit(train_generator, validation_data = validation_generator,epochs=50, steps_per_epoch=steps,callbacks=callbacks) Epoch 1/50 Epoch 2/50 In [27]: print("Accuracy->", (history_1.history['accuracy'][-1])*100)

1. Download all the data in this folder https://drive.google.com/open?id=1Z4TyI7FcFVEx8gdl4j09gxvxaqLSgoEu. it contains two file both

2. On this image data, you have to train 3 types of models as given below. You have to split the data into Train and Validation data.

https://medium.com/@vijayabhaskar96/tutorial-on-keras-imagedatagenerator-with-flow-from-dataframe-8bd5776e45c1

https://medium.com/@vijayabhaskar96/tutorial-on-keras-flow-from-dataframe-1fd4493d237c

don't include histograms, check the documentation of tensorboard for more information.

4. Train only new Conv block, FC layers, output layer. Don't train the VGG-16 network.

3. Try not to load all the images into memory, use the gernarators that we have given the reference notebooks to load the batch of images

4. You are free to choose Learning rate, optimizer, loss function, image augmentation, any hyperparameters. but you have to use the same

Note: fit_genarator() method will have problems with the tensorboard histograms, try to debug it, if you could not do use histgrams=0 i.e

6. You can check about Transfer Learning in this link - https://blog.keras.io/building-powerful-image-classification-models-using-very-

2. After VGG-16 network without FC layers, add a new Conv block (1 Conv layer and 1 Maxpooling), 2 FC layers and a output layer to

3. Final architecture will be INPUT --> VGG-16 without Top layers(FC) --> Conv Layer --> Maxpool Layer --> 2 FC layers --> Output Layer

5. Use tensorboard for every model and analyse your gradients. (you need to upload the screenshots for each model for evaluation)

1. Use VGG-16 pretrained network without Fully Connected layers and initilize all the weights with Imagenet trained weights.

classify 16 classes. You are free to choose any hyperparameters/parameters of conv block, FC layers, output layer.

images and labels. The label file list the images and their categories in the following format:

path/to/the/image.tif,category

0 letter 1 form 2 email

3 handwritten 4 advertisement 5 scientific report

7 specification 8 file folder 9 news article

12 presentation 13 questionnaire

only during the train data. or you can use this method also

architechture what we are asking below.

UsageError: Line magic function `%tensorflow_version` not found.

!pip install git+https://github.com/keras-team/keras-preprocessing.git

10 budget 11 invoice

14 resume 15 memo

little-data.html

%tensorflow_version 2.x

import tensorflow as tf

Accuracy-> 60.583335161209106

%tensorboard --logdir='logs/fit/'

ERROR

Generated Sat, 04 Sep 2021 06:57:57 GMT by htmlpdf-proxy (squid/4.10)

Reusing TensorBoard on port 6006 (pid 444), started 1:00:53 ago. (Use '!kill 444' to kill it.)

Access control configuration prevents your request from being allowed at this time. Please contact your service provider if you feel this is incorrect.

1. Use VGG-16 pretrained network without Fully Connected layers and initilize all the weights with Imagenet trained weights.

2. After VGG-16 network without FC layers, don't use FC layers, use conv layers only as Fully connected layer. any FC layer can be

depth column "fits" across the input volume, giving identical result as the initial FC layer. You can refer this link to better

classification. INPUT --> VGG-16 without Top layers(FC) --> 2 Conv Layers identical to FC --> Output Layer 3. Train only last 2 Conv layers identical to FC layers, 1 output layer. Don't train the VGG-16 network.

base_model = tf.keras.applications.vgg16.VGG16(weights='imagenet', include_top=False, input_shape=(*target_size, 3))

Param #

1792

36928

73856

147584

295168

590080

590080

1180160

2359808

2359808

2359808

2359808

2359808

2359808

589952

8208

tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1, write_graph=True, write_grads=True)

history_2 = model_2.fit(train_generator, validation_data=validation_generator,epochs=50, steps_per_epoch=steps,callbacks=callbacks)

1. Use same network as Model-2 'INPUT --> VGG-16 without Top layers(FC) --> 2 Conv Layers identical to FC --> Output Layer' and train only

('You must install pydot (`pip install pydot`) and install graphviz (see instructions at https://graphviz.gitlab.io/download/) ', 'for plot_model_model_to_dot

0

converted to a CONV layer. This conversion will reduce the No of Trainable parameters in FC layers. For example, an FC layer with K=4096 that is looking at some input volume of size 7×7×512 can be equivalently expressed as a CONV layer with F=7,P=0,S=1,K=4096. In other words, we are setting the filter size to be exactly the size of the input volume, and hence the output will simply be 1×1×4096 since only a single

3. Final architecture will be VGG-16 without FC layers(without top), 2 Conv layers identical to FC layers, 1 output layer for 16 class

The requested URL could not be retrieved

understanding of using Conv layer in place of fully connected layers.

Conv2D_1 = Conv2D(filters = 512, kernel_size=(3, 3), activation='relu')(base_model.output)

model_2.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])

= Dense(units = 16, activation='softmax')(flatten)

Output Shape

[(None, 128, 128, 3)]

(None, 128, 128, 64)

(None, 128, 128, 64)

(None, 64, 64, 64)

(None, 64, 64, 128)

(None, 64, 64, 128)

(None, 32, 32, 128)

(None, 32, 32, 256)

(None, 32, 32, 256)

(None, 32, 32, 256)

(None, 16, 16, 256)

(None, 16, 16, 512)

(None, 16, 16, 512)

(None, 16, 16, 512)

(None, 8, 8, 512)

(None, 8, 8, 512)

(None, 8, 8, 512)

(None, 8, 8, 512)

(None, 4, 4, 512)

(None, 2, 2, 512)

(None, 2, 2, 128)

log_dir ="logs/fit/"+"model_2_" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")

WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.

Reusing TensorBoard on port 6006 (pid 444), started 10:17:14 ago. (Use '!kill 444' to kill it.)

Access control configuration prevents your request from being allowed at this time. Please contact your service provider if you feel this is incorrect.

Last 6 Layers of VGG-16 network, 2 Conv layers identical to FC layers, 1 output layer.

Conv2D_1 = Conv2D(filters = 512, kernel_size=(3, 3), activation='relu')(base_model.output)

model_3.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['accuracy'])

Conv2D_2 = Conv2D(filters = 128, kernel_size=(3, 3), activation='relu', padding='same')(Conv2D_1)

base_model = tf.keras.applications.vgg16.VGG16(weights='imagenet', include_top=False, input_shape=(*target_size, 3))

Param #

1792

36928

73856

147584

295168

590080

590080

1180160

2359808

2359808

2359808

2359808

2359808

2359808

589952

8208

tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir, histogram_freq=1, write_graph=True, write_grads=True)

history_3 = model_3.fit(train_generator, validation_data=validation_generator,epochs=50, steps_per_epoch=steps,callbacks=callbacks)

('You must install pydot (`pip install pydot`) and install graphviz (see instructions at https://graphviz.gitlab.io/download/) ', 'for plot_model_to_dot

0

(None, 512)

(None, 16)

early_stoping = keras.callbacks.EarlyStopping(patience=2)

callbacks = [early_stoping, tensorboard_callback, target]

print("Accuracy->", (history_2.history['accuracy'][-1])*100)

The requested URL could not be retrieved

The following error was encountered while trying to retrieve the URL: http://127.0.0.1:6006/

model_2 = Model(inputs=base_model.inputs, outputs=Out)

callbacks = [early_stoping, tensorboard_callback, target]

steps = len(train_generator.labels) // batch_size

Conv2D_2 = Conv2D(filters = 128, kernel_size=(3, 3), activation='relu', padding='same')(Conv2D_1)

The following error was encountered while trying to retrieve the URL: http://127.0.0.1:6006/

%reload_ext tensorboard

Access Denied.

Model-2

tf.keras.backend.clear_session()

for layer in base_model.layers: layer.trainable = False

flatten = Flatten()(Conv2D_2)

target = AchieveAccuracy(0.6)

print(model_2.summary())

np.random.seed(0)

batch_size = 256

Model: "model"

input_1 (InputLayer)

block1_conv1 (Conv2D)

block1_conv2 (Conv2D)

block2_conv1 (Conv2D)

block2_conv2 (Conv2D)

block3_conv1 (Conv2D)

block3_conv2 (Conv2D)

block3_conv3 (Conv2D)

block4_conv1 (Conv2D)

block4_conv2 (Conv2D)

block4_conv3 (Conv2D)

block5_conv1 (Conv2D)

block5_conv2 (Conv2D)

block5_conv3 (Conv2D)

conv2d (Conv2D)

conv2d_1 (Conv2D)

flatten (Flatten)

dense (Dense)

to work.')

Epoch 1/50

Epoch 2/50

Epoch 3/50

Epoch 4/50

Epoch 5/50

Epoch 6/50

Accuracy-> 60.50000190734863

%tensorboard --logdir='logs/fit/'

ERROR

Generated Sat, 04 Sep 2021 06:57:57 GMT by htmlpdf-proxy (squid/4.10)

%reload_ext tensorboard

Access Denied.

Model-3

%%time

batch_size = 256

Wall time: 1.03 ms

np.random.seed(0)

rn.seed(0)

target = AchieveAccuracy(0.065)

tf.keras.backend.clear_session()

layer.trainable = False

flatten = Flatten()(Conv2D_2)

print(model_3.summary())

Model: "model"

input_1 (InputLayer)

block1_conv1 (Conv2D)

block1_conv2 (Conv2D)

block2_conv1 (Conv2D)

block2_conv2 (Conv2D)

block3_conv1 (Conv2D)

block3_conv2 (Conv2D)

block3_conv3 (Conv2D)

block4_conv1 (Conv2D)

block4_conv2 (Conv2D)

block4_conv3 (Conv2D)

block5_conv1 (Conv2D)

block5_conv2 (Conv2D)

block5_conv3 (Conv2D)

conv2d (Conv2D)

conv2d_1 (Conv2D)

flatten (Flatten)

dense (Dense)

to work.')

Epoch 1/50

Epoch 2/50

Accuracy-> 6.77083358168602

%tensorboard --logdir='logs/fit/'

ERROR

Generated Sat, 04 Sep 2021 06:57:57 GMT by htmlpdf-proxy (squid/4.10)

%reload_ext tensorboard

Access Denied.

Your cache administrator is <u>webmaster</u>.

In [40]:

In [41]:

In [29]:

In [42]:

In []:

block4_pool (MaxPooling2D)

block5_pool (MaxPooling2D)

Total params: 17,672,656 Trainable params: 12,397,200 Non-trainable params: 5,275,456

block1_pool (MaxPooling2D)

block2_pool (MaxPooling2D)

block3_pool (MaxPooling2D)

Layer (type)

for layer in base_model.layers[:-6]:

Out = Dense(16, activation='softmax')(flatten)

model_3 = Model(inputs=base_model.inputs, outputs=Out)

Output Shape

[(None, 128, 128, 3)]

(None, 128, 128, 64)

(None, 128, 128, 64)

(None, 64, 64, 64)

(None, 64, 64, 128)

(None, 64, 64, 128)

(None, 32, 32, 128)

(None, 32, 32, 256)

(None, 32, 32, 256)

(None, 32, 32, 256)

(None, 16, 16, 256)

(None, 16, 16, 512)

(None, 16, 16, 512)

(None, 16, 16, 512)

(None, 8, 8, 512)

(None, 8, 8, 512)

(None, 8, 8, 512)

(None, 8, 8, 512)

(None, 4, 4, 512)

(None, 2, 2, 512)

(None, 2, 2, 128)

log_dir ="logs/fit/"+"model_3_" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")

WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.

Reusing TensorBoard on port 6006 (pid 444), started 12:48:52 ago. (Use '!kill 444' to kill it.)

Access control configuration prevents your request from being allowed at this time. Please contact your service provider if you feel this is incorrect.

(None, 512)

(None, 16) ______

early_stoping = keras.callbacks.EarlyStopping(patience=2)

print("Accuracy->", (history_3.history['accuracy'][-1])*100)

The requested URL could not be retrieved

The following error was encountered while trying to retrieve the URL: http://127.0.0.1:6006/

steps = len(train_generator.labels) // batch_size

callbacks = [early_stoping, tensorboard_callback, target]

In [37]:

In [38]:

In [39]:

Your cache administrator is webmaster.

In [19]:

In [20]:

In [21]:

In [22]:

block5_pool (MaxPooling2D)

Total params: 17,672,656 Trainable params: 2,957,968 Non-trainable params: 14,714,688

block1_pool (MaxPooling2D)

block2_pool (MaxPooling2D)

block3_pool (MaxPooling2D)

block4_pool (MaxPooling2D)

Layer (type)

rn.seed(0)

In [17]:

In [18]:

Your cache administrator is webmaster.

In [28]:

import os

!pip uninstall keras-preprocessing

Model-1

In [1]:

In [3]:

6 scientific publication

where the categories are numbered 0 to 15, in the following order: